パターン認識
Pattern Recognition

佐藤真一
Shin'ichi Satoh

国立情報学研究所
National Institute of Informatics

May 23, 2023

本講義について

- 講義のページ: https://research.nii.ac.jp/~satoh/utpr/
- 講義資料は上記のページにあり
- 講義映像は ITC-LMS にあり
- 単位は最終レポートと宿題によって評価予定 (最終レポートの提出は必須、宿題は全 7 回のうち 3 回の提出必須)
- 出席は取らない

スケジュール (予定)

4/11		講義概要、ベイズ決定測、確率分布
4/18		確率変数、確率変数ベクトル、正規分布
4/25		確率分布パラメータ推定、識別関数
5/2		ノンパラメトリック分布推定、パルゼン窓、k 近傍推定
5/9		k 近傍識別器、識別誤り推定
5/16		ベイズ誤り推定、識別誤り推定、交差判定法、ブートストラップ法
5/23	ハイブリッド	線形識別器、パーセプトロン、最小自乗誤差 (MSE) 識別器、Widrow-Hoff 則
5/30	オンライン (zoom) のみ	ニューラルネットワーク、深層学習
6/6	ハイブリッド	サポートベクターマシン
6/13	オンライン (zoom) のみ	直交展開、固有値展開
6/20		休講
6/27	ハイブリッド	クラスタリング、デンドログラム、凝集型 (aggromerative) クラスタリング、k-means
7/4	ハイブリッド	グラフ、ノーマライズドカット、スペクトラルクラスタリング、ラプラシアンアイゲンマップ
7/11		予備日

本日の内容

- 線形識別器 / 線形識別関数
- パーセプトロン

識別関数

与えられた観測を c のうちの一つのクラスに割り当てる問題を考える
この問題は識別関数を使って定式化できる: $g_i(x)$, $i = 1, \ldots, c$
識別器は特徴ベクトル x を以下の場合にクラス ω_i に識別する:

$$g_i(x) > g_j(x) \text{ for all } j \neq i.$$

線形識別関数

入力を d 次元ベクトルであると仮定する:

$$x = [x_1\ x_2\ \cdots\ x_d]^T$$

また重みベクトルを考える:

$$w = [w_1\ w_2\ \cdots\ w_d]^T.$$

線形識別関数は以下により定義される

$$g(x) = w_0 + \sum_{j=1}^{d} w_j x_j$$

線形識別関数

拡張特徴ベクトルと拡張重みベクトルを考える:

$$\hat{x} = [1\, x_1\, x_2\, \cdots\, x_d]^T$$
$$\hat{w} = [w_0\, w_1\, w_2\, \cdots\, w_d]^T.$$

線形識別関数は以下のようになる:

$$\begin{aligned} g(x) &= w_0 + \sum_{j=1}^{d} w_j x_j \\ &= w_0 + w^T x \\ &= \hat{w}^T \hat{x}. \end{aligned}$$

(以下、自明な場合には w と $\hat{w}$, x と $\hat{x}$, 重みベクトル/拡張重みベクトル, 特徴ベクトル/拡張特徴ベクトルを区別せずに使う)

線形識別関数

どのようなときに線形識別関数が使われるか?
パラメトリック確率分布推定において、$\Sigma_i = \sigma^2 I$ の時
この時の識別関数は:

$$\begin{aligned} g_i(x) &= -\frac{||x - \mu_i||^2}{2\sigma^2} + \log P(\omega_i) \\ &= -\frac{1}{2\sigma^2}[x^T x - 2\mu_i^T x + \mu_i^T \mu_i] + \log P(\omega_i). \end{aligned}$$

共通項を省略することにより線形識別関数が得られる:

$$g_i(x) = w_i^T x + w_{i0}$$

ただし

$$w_i = \frac{1}{\sigma^2}\mu_i \text{ and } w_{i0} = -\frac{1}{2\sigma^2}\mu_i^T \mu_i + \log P(\omega_i).$$

線形識別関数

最近傍識別器も別の事例となる
n 個のプロトタイプがあるとする: $p_1, p_2, \cdots, p_n$
最近傍識別則は、入力ベクトル x に最も近いプロトタイプを選ぶ:

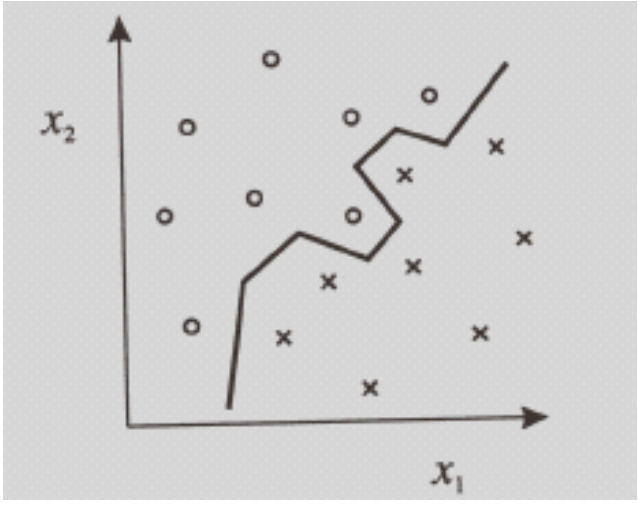
$$\|x - p_i\|^2 = \|x\|^2 - 2p_i^T x + \|p_i\|^2.$$

従って、識別関数は:

$$g_i(x) \stackrel{\text{def}}{=} p_i^T x - \frac{1}{2}\|p_i\|^2.$$

識別境界は区分線形となる

線形識別関数



線形識別関数

与えられた学習データよりどのように線形識別器を求めるか?
学習データの集合 $\mathcal{X}$ を考え、各クラス ω_i に対応する学習データの集合を $\mathcal{X}_i$ $(i = 1, 2, \cdots, c)$ とする
線形識別関数の学習とは、$\mathcal{X}_i$ 中の全サンプルについて、以下が満足されるように拡張重みベクトル $\hat{w}_i$ を決定することとなる:

$$g_i(x) > g_j(x) \text{ for all } j \neq i$$

そのような拡張重みベクトル $\hat{w}_i$ が存在する場合、$\mathcal{X}$ は線形分離可能という

2 クラスの場合について考えよう: ω_1, ω_2
その場合問題は以下のように簡略化できる

$$\begin{aligned} g(x) = g_1(x) - g_2(x) &= (w_1 - w_2)^T x \\ &= w^T x \\ w &\stackrel{\text{def}}{=} w_1 - w_2 \end{aligned}$$

かつ

decide ω_1 if $g(x) = w^T x > 0$

decide ω_2 if $g(x) = w^T x < 0$.

クラス ω_2 に属するすべてのサンプルに負号をつけることにより、データを正規化できる
この場合すべてのデータについて $g(x) = w^T x > 0$ となる

パーセプトロン

Rosenblatt, 1957[1]
全学習サンプルについて $w^T x > 0$ となるように w を決定したい
パーセプトロン損失関数:

$$J(w) = \sum_{x \in \tilde{\mathcal{X}}} (-w^T x)$$

ただし $\tilde{\mathcal{X}}$ は識別誤りとなっているすべてのサンプル
J は常に非負であり、これをゼロにしたい (すなわち、$\tilde{\mathcal{X}}$ を空集合としたい)

[1] F. Rosenblatt, The perceptron - A perceiving and recognizing automaton, Technical Report 85-460-1, Cornell Aeronautical Laboratory, Ithaca, New York, January, 1957.

J の勾配 (gradient) は

$$\nabla J = \sum_{x \in \tilde{\mathcal{X}}} (-x)$$

なので、最急降下法 (勾配降下法, Gradient Descent, もしくは batch Gradient Descent) により重みベクトルを更新可能

$$w(k+1) = w(k) - \rho \nabla J = w(k) + \rho \sum_{x \in \tilde{\mathcal{X}}} x.$$

ρ は学習率 (learning rate) と呼ばれる

パーセプトロン

Algorithm 1 Batch Perceptron

1: Initialization: w,ρ, criterion θ, $k = 0$
2: **repeat**
3: $\quad k \leftarrow k + 1$
4: $\quad w \leftarrow w + \rho \sum_{\tilde{x} \in \tilde{\mathcal{X}}} \tilde{x}$
5: $\quad$ Recalculate $\tilde{\mathcal{X}}$
6: **until** $|\rho \sum_{\tilde{x} \in \tilde{\mathcal{X}}} \tilde{x}| < \theta$
7: Return w

確率的最急降下法 (Stocastic Gradient Descent, on-line Gradient Descent) も利用可能

Algorithm 2 Fixed-Increment Single-Sample Perceptron

1: Initialization: w, $\tilde{\mathcal{X}} = \{\tilde{x_1}, \tilde{x_2}, \cdots\}$, $k = 0$
2: **repeat**
3: $k \leftarrow k + 1$
4: $i \leftarrow \mathrm{mod}(k, |\tilde{\mathcal{X}}|) + 1$
5: $w \leftarrow w + \rho \tilde{x_i}$
6: Recalculate $\tilde{\mathcal{X}}$
7: **until** all patterns properly classified
8: Return w

パーセプトロン

学習データが線形分離可能なとき、パーセプトロンは収束することが証明されている

最小二乗誤差法と疑似逆行列

n 個の学習データと仮定

$$\mathcal{X} = \{x_1, x_2, \cdots, x_n\}.$$

p 番目のデータを x_p とし、c 個の識別関数の出力をベクトルとしてとらえる:

$$[g_1(x_p)\, g_2(x_p)\, \cdots\, g_c(x_p)]^T.$$

さらに目的の値のベクトルを仮定する:

$$b_p = [b_{1p}\, b_{2p}\, \cdots\, b_{cp}]^T$$

以下であることに注意: $b_{ip} > b_{jp}\ (j \neq i)$ if $x_p \in \mathcal{X}_i$.
例えば、$x_p \in \mathcal{X}_i$ について

$$b_p = [0 \cdots 0 \underset{i}{1} 0 \cdots 0]^T$$

$g_i(x_p) \approx b_{ip}$ となるように w_p を決めたい

最小二乗誤差法と疑似逆行列

パターン x_p に関する誤差は $\varepsilon_{ip} = g_i(x_p) - b_{ip}$
損失関数を以下の通り定義する:

$$
\begin{aligned}
J_p(\hat{w}_1, \hat{w}_2, \cdots, \hat{w}_c) &= \frac{1}{2} \sum_{i=1}^{c} \varepsilon_{ip}^2 \\
&= \frac{1}{2} \sum_{i=1}^{c} (g_i(x_p) - b_{ip})^2 \\
&= \frac{1}{2} \sum_{i=1}^{c} (\hat{w_i}^T \hat{x_p} - b_{ip})^2
\end{aligned}
$$

最小二乗誤差法と疑似逆行列

バッチ損失関数:

$$
\begin{aligned}
J(w_1, w_2, \cdots, w_c) &= \sum_{p=1}^{n} J_p(w_1, w_2, \cdots, w_c) \\
&= \frac{1}{2} \sum_{p=1}^{n} \sum_{i=1}^{c} (g_i(x_p) - b_{ip})^2 \\
&= \frac{1}{2} \sum_{p=1}^{n} \sum_{i=1}^{c} (\hat{w_i}^T \hat{x_p} - b_{ip})^2
\end{aligned}
$$

損失関数を最小化するように w_i を決めたい
または

$$
\left[\hat{w_1}\ \hat{w_2}\ \cdots\ \hat{w_c}\right]^T \left[\hat{x_1}\ \hat{x_2}\ \cdots\ \hat{x_n}\right] \approx \begin{bmatrix} & \vdots & \\ \cdots & b_{ip} & \cdots \\ & \vdots & \end{bmatrix}
$$

最小二乗誤差法と疑似逆行列

2 クラスの場合:

$$
\begin{aligned}
J_p(\hat{w}) &= \frac{1}{2}(g(x_p) - b_p)^2 \\
&= \frac{1}{2}(\hat{w}^T \hat{x_p} - b_p)^2
\end{aligned}
$$

b_p として以下を考える

$$
b_p = \begin{cases} 1 & (x_p \in \mathcal{X}_1) \\ -1 & (x_p \in \mathcal{X}_2) \end{cases}
$$

最小二乗誤差法と疑似逆行列

J の最小化は、疑似逆行列を用いて閉形式 (closed-form) で与えられる:

$$\nabla J = \frac{\partial J}{\partial \hat{w}} = [\frac{\partial J}{\partial w_0} \frac{\partial J}{\partial w_1} \cdots \frac{\partial J}{\partial w_d}]$$

J は以下により最小化できる:

$$\frac{\partial J}{\partial \hat{w}_i} = \nabla_i J = 0 \ (i = 1, \cdots, c)$$

すなわち

$$\begin{aligned} \frac{\partial J}{\partial \hat{w}_i} &= \sum_{p=1}^{n} \frac{\partial J_p}{\partial \hat{w}_i} \\ &= \sum_{p=1}^{n} (\hat{w}_i{}^T \hat{x_p} - b_{ip}) \hat{x_p} = 0 \end{aligned}$$

最小二乗誤差法と疑似逆行列

以下とすると

$$X = [\hat{x_1}\ \hat{x_2}\ \cdots\ \hat{x_n}]^T$$
$$b_i = [b_{i1}\ b_{i2}\ \cdots\ b_{in}]^T\ (i = 1, \cdots, c)$$

損失関数は

$$J(\hat{w_1}, \hat{w_2}, \cdots, \hat{w_c}) = \frac{1}{2}\sum_{i=1}^{c} \|X\hat{w_i} - b_i\|^2$$
$$\frac{\partial J}{\partial \hat{w_i}} = X^T(X\hat{w_i} - b_i) = 0$$
$$X^T X\hat{w_i} = X^T b_i$$
$$\hat{w_i} = (X^T X)^{-1} X^T b_i$$

これは $\|Xw_i - b_i\|^2$ の最小二乗 (MSE) 解を与える

最小二乗誤差法と疑似逆行列

$X^+ = (X^T X)^{-1} X^T$ は X の疑似逆行列と呼ばれる

ここで最急降下法を考える

$$\hat{w}_i(k+1) = \hat{w}_i(k) - \rho \frac{\partial J}{\partial \hat{w}_i} = \hat{w}_i(k) - \rho \nabla_i J$$

$$\Delta \hat{w}_i = -\rho \nabla_i J$$

確率的最急降下法を考えてもよい

$$\Delta \hat{w}_i = -\rho \frac{\partial J_p}{\partial \hat{w}_i}$$

$g_i(x_p)$ を g_{ip} と記載すると

$$\frac{\partial J_p}{\partial \hat{w}_i} = \frac{\partial J_p}{\partial g_{ip}} \frac{\partial g_{ip}}{\partial \hat{w}_i}$$

ただし

$$\frac{\partial J_p}{\partial g_{ip}} = g_{ip} - b_{ip} = \varepsilon_{ip}$$
$$\frac{\partial g_{ip}}{\partial \hat{w}_i} = \hat{x_p}$$

従って

$$\frac{\partial J_p}{\partial \hat{w}_i} = (g_{ip} - b_{ip})\hat{x_p} = \varepsilon_{ip}\hat{x_p}$$

Widrow-Hoff 法/LMS 法

更新式は以下の通り

$$\begin{aligned}\Delta \hat{w}_i &= -\rho \varepsilon_{ip} \hat{x_p} \\ &= -\rho (g_{ip} - b_{ip}) \hat{x_p} \\ &= -\rho ({\hat{w}_i}^T \hat{x_p} - b_{ip}) \hat{x_p}.\end{aligned}$$

これを Widrow-Hoff 法 もしくは LMS 法 (least-mean-squared) と呼ぶ

Widrow-Hoff 法/LMS 法

Algorithm 3 LMS

1: Initialization: $\hat{w}_i$, $k = 0$
2: **repeat**
3: $\quad k \leftarrow \mathrm{mod}(k, |\mathcal{X}|) + 1$
4: $\quad \hat{w}_i \leftarrow \hat{w}_i - \rho(\hat{w}_i{}^T \hat{x}_k - b_{ip})\hat{x}_k$
5: **until** all patterns properly classified
6: Return $\hat{w}_i$

宿題

- プログラミング課題と非プログラミング課題を課する
- プログラミング課題か非プログラミング課題かのいずれかを解いて提出すること
- プログラミング課題を解くことを推奨する
- もちろん両方解いてもらえると嬉しい
- 締切は 6/6

プログラミング課題

以下の Matlab プログラムにより三種類の学習データを生成せよ

Linearly separable linear.m

Linearly non-separable nonlinear.m

Skewed linearly separable slinear.m

Python の場合、以下の行をプログラムに加え三種類の学習データを生成せよ

Linearly separable from linear import *

Linearly non-separable from nonlinear import *

Skewed linearly separable from slinear import *

(1) 拡張特徴ベクトル、拡張重みベクトルと、正規化を実装し、パーセプトロン (バッチもしくはオンライン) を実装せよ。三種類のデータでパーセプトロンを実行せよ。その振る舞いについて論ぜよ
ヒント: 拡張特徴ベクトルは以下により実現可能

```
ax = np.concatenate((np.ones((1, n)), x))
```

(2) MSE 法による識別器を実装し、三種類のデータにより学習させよ。その振る舞いについて論ぜよ

プログラミング課題 (1)

```
import numpy as np
import matplotlib.pyplot as plt
from linear import *

rho = 0.1
ax = np.concatenate((np.ones((1, n)), x))
aw = (2 * np.random.rand(d + 1) - np.array([1, 1, 1]))[:, np.newaxis]
ax[:, np.where(l == -1)] = -ax[:, np.where(l == -1)]
plt.figure()
k = 0
neg = ((ax.T.dot(aw)).T < 0)[-1]
```

プログラミング課題 (1)

```
while len(np.where(neg)[-1]) > 0:
    k += 1
    aw += rho*<<< some code to update aw >>>
    neg = <<< some code to update neg >>>
    plt.clf()
    plt.xlim([-1, 1])
    plt.ylim([-1, 1])
    plt.plot(x[0, np.where((l == 1) & ~neg)],
             x[1, np.where((l == 1) & ~neg)], 'bo')
    plt.plot(x[0, np.where((l == -1) & ~neg)],
             x[1, np.where((l == -1) & ~neg)], 'bx')
    plt.plot(x[0, np.where((l == 1) & neg)],
             x[1, np.where((l == 1) & neg)], 'ro')
    plt.plot(x[0, np.where((l == -1) & neg)],
             x[1, np.where((l == -1) & neg)], 'rx')
```

プログラミング課題 (1)

```
    if abs(aw[1]) > abs(aw[2]):
        plt.plot([-1, 1], [-(aw[0] - aw[1]) / aw[2], -(aw[0] + aw[1]) / aw[2]])
    else:
        plt.plot([-(aw[0] - aw[2]) / aw[1], -(aw[0] + aw[2]) / aw[1]], [-1, 1])
    print(aw)
    plt.pause(0.2)
plt.show()
```

プログラミング課題 (2)

```
import numpy as np
import matplotlib.pyplot as plt
from linear import *

ax = np.concatenate((np.ones((1, n)), x))
aw = <<< some code to compute aw >>>
neg = (ax.T.dot(aw)).T < 0
# Similar code to perceptron follows...
```

Similar code to perceptron follows...

非プログラミング課題

(1) パーセプトロン収束定理を証明せよ (バッチもしくはオンライン)
(2) 最小自乗誤差法の解が疑似逆行列により求められることを示せ。すなわち、以下の時

$$J_p(\hat{w}_1, \hat{w}_2, \cdots, \hat{w}_c) = \frac{1}{2}\sum_{i=1}^{c}(\hat{w}_i{}^T \hat{x}_p - b_{ip})^2$$

$\frac{\partial J}{\partial \hat{w}_i} = 0$ の解が $\hat{w}_i = X^+ b_i$ であることを示せ